Towards AutoML in the presence of Drift: first results

Jorge Madrid, Hugo Jair Escalante, Eduardo Morales, Wei-Wei Tu, Yang Yu, Lisheng Sun-Hosoya, Isabelle Guyon, Michèle Sebag

Towards AutoML in the presence of Drift: first results

Jorge G. Madrid JORGEGUS.93@GMAIL.COM
INAOE, Mexico
Hugo Jair Escalante HUGOJAIR@INAOEP.MX
INAOE, Mexico
Eduardo F. Morales EMORALES@INAOEP.MX
INAOE, Mexico
Wei-Wei Tu TUWWCN@GMAIL.COM
4Paradigm Inc., China
Yang Yu YUY@NJU.EDU.CN
Nanjing University, China
Lisheng Sun-Hosoya CECILE829@GMAIL.COM
UPSud, U. Paris-Saclay, France
Isabelle Guyon GUYON@CHALEARN.ORG
UPSud/INRIA, U. Paris-Saclay, France and ChaLearn, USA
Michele Sebag MICHELE.SEBAG@LRI.FR
CNRS, U. Paris-Saclay, France

Abstract

Research progress in AutoML has lead to state of the art solutions that can cope quite well with supervised learning task, e.g., classification with AutoSklearn. However, so far these systems do not take into account the changing nature of evolving data over time (i.e., they still assume i.i.d. data); even when this sort of domains are increasingly available in real applications (e.g., spam filtering, user preferences, etc.). We describe a first attempt to develop an AutoML solution for scenarios in which data distribution changes relatively slowly over time and in which the problem is approached in a lifelong learning setting. We extend Auto-Sklearn with sound and intuitive mechanisms that allow it to cope with this sort of problems. The extended Auto-Sklearn is combined with concept drift detection techniques that allow it to automatically determine when the initial models have to be adapted. We report experimental results in benchmark data from AutoML competitions that adhere to this scenario. Results demonstrate the effectiveness of the proposed methodology.

Keywords: AutoML, Life Long Machine Learning, Concept Drift, AutoSKLearn,

1. Introduction

Autonomous Machine Learning (AutoML) is the field focusing on methods that aim at automating different stages of the machine learning process. AutoML solutions are increasingly receiving more attention from both the ML community and users because of (1) the large amounts of data readily available everywhere, and (2) the lack of domain and/or ML experts who can advise/supervise the development of ML-based systems.

Although progress in AutoML is vast, the considered scenarios are somewhat constrained, e.g., in the type of approached problem, in the assumptions on data, in the size

of datasets, etc. In this context, one of the most desirable features for AutoML methods is to work under a *lifelong machine learning* (LML) setting. LML refers to systems that can sequentially learn many tasks from one or more domains in its lifetime (Silver and amd Lianghao Li, 2013), these systems (not restricted to supervised learning) require the ability to retain knowledge, adapt to changes and transfer knowledge when learning a new task. An AutoML method that learns from different tasks and that is able to adapt itself during its lifetime would comprise a competitive and robust all-problem machine learning solution.

This paper aims at exploring the viability of AutoML methods to operate in a LML setting, in particular in a scenario where the targets evolve over time, that is, in the presence of concept drift. We modify the Auto-Sklearn method with mechanisms that allow it to deal with the drift phenomenon in a simplified LML evaluation scenario. The proposed mechanisms are sound and highly intuitive, yet very useful. We perform experiments in benchmark data from concept drift and AutoML with drift. Experimental results reveal that the proposed mechanisms allow Auto-Sklearn to successfully cope with drift. To the best of our knowledge these are the first results reported on AutoML for a simplified LML scenario in the presence of concept drift[1].

2. Related work

Although the term 'AutoML' was coined recently by F. Hutter and collaborators (Thornton et al., 2013; Hutter, 2009), the problem of hyper-parameter selection has been studied for several decades in the machine learning community (Bozdogan, 1987; Bengio, 2000; Bergstra et al., 2011; Bergstra and Bengio, 2012). However, the original emphasis was on over-fitting avoidance while, with the emergence of "big data" the current emphasis is on search efficiency. Novel effective approaches have been proposed in the academic literature that have become wide spread among practitioners because they are both theoretically well founded and practically efficient: (1) Bayesian Optimization (BO) methods build a posterior $p(model|data)$ by applying candidate models to the input data and use this posterior distribution to guide the search (e.g. (Hutter et al., 2011; Swersky et al., 2014)). (2) Complementary to BO, Meta Learning develops a set of meta-features capturing the nature of data, which are then used to infer the model performance based on past experiences on similar data, without actually training the model (e.g. (Muñoz et al., 2018)). (3) Evolutionary Algorithms (EA) learn a distribution over hyper-parameters and updates it to help the search (e.g. (Real et al., 2017)). (4) Reinforcement learning approaches (RL), where the hyper-parameter optimization problem is formulated as learning an efficient policy to move in the hyper-parameter space and solved using RL techniques (e.g. (Zoph and Le, 2016; Baker et al., 2016)). The algorithm Auto-sklearn used in this paper is based on a BO approach initialized with Meta Learning.

Hyper-parameter selection usually focuses on solving tasks in isolation, but, if tasks show some sequential dependency, a lifelong learning (LML) approach can be applied to continuously retain and immigrate knowledge across tasks and make the future learning

1. One should note that our work is different from standard data-stream / concept drift learning methods in the sense that in AutoML we aim to autonomously find preprocessing, feature selection and classification methods (together with hyper parameter optimization), while in most of the existing literature the problem is reduced to modify the classification model.

more efficient. The concept of LML was first introduced in robot control (Thrun and Mitchell, 1995), where, inspired by the fact that a robot might be exposed to various learning tasks during their lifetime and the knowledge learned might be transfered to make future learning more efficient, they proposed EBNN (explanation-based neural network): suppose the robot perform all tasks in the same environment (e.g. housekeeping robot), then its knowledge about the environment (called *action model*, the mapping between state-action pairs to the next state) can be transfered to new task using back-propagation. Since then, diverse techniques have been developed in lifelong supervised learning (Silver and Mercer, 2002, 1996, 2001; Chen et al., 2018; Fei et al., 2016).

Due to the big volume of data being constantly generated, many prediction problems need a model that continuously receives data and thus cannot work in an off-line/static mode with historical data. Some of these environments are non-stationary where data distributions change over time, this phenomenon is known as concept drift (Schlimmer and Granger, 1986; Widmer and Kubat, 1996; Gama et al., 2014). Numerous research work have been done for adapting models to data streams with presence of concept drift. Hulten et al. (2001) developed CVFDT where a Decision Tree is maintained up-to-date with regard to a sliding window of examples: an alternative sub-tree is built when drift is detected in the window, and is used to replace the old sub-tree when this new one becomes more performing. Then, by proving the out-performance of ensembling compared to single classifiers in the concept drift environment, Wang et al. (2003b) proposed weighted classifier ensemble, where adaptation is done by dynamically updating the weight for each base classifier according to its expected prediction accuracy on current test stream. To address the problem of sliding window size, Bifet and Gavalda (2007) introduced Adwin in which the window is resized based on changing rate observed in the window, Adwin is also used widely as a change detector as in Bifet et al. (2010) and van Rijn et al. (2014). The same authors then came up with Leverage Bagging (Bifet et al. (2010)) where examples are weighted by a well parameterized Poisson distribution to add randomization in input data that improve the accuracy of ensembling. van Rijn et al. (2014), Rossi et al. (2014) and van Rijn et al. (2015) collaborated meta-learning techniques to solve the algorithm selection problem: predict best model for next upcoming stream based on meta-knowledge collected from previous streams.

Although work for processing streams and data in the presence of drift is vast, to the best of our knowledge existing work as not approaching the AutoML setting: automatically building and updating a *full model* (Escalante et al., 2009), that is a model that comprises data preprocessing, feature selection and classification model and that optimizes hyperparameters of the whole model.

In the scenario of LML-AutoML, the learning algorithm should be able to incorporate new data and update previous predictive models. In this paper, we follow the 3 steps of on-line adaptive learning procedure proposed in (Gama et al., 2014): (1)*Predict*. When new example X_t arrives, a prediction $\hat{y}t$ is made using the current model L_t. (2)*Diagnose*. After some time the true label y_t is received and the loss can be estimated as $f(\hat{y}_t, y_t)$, and (3)*Update*. use the example (X_t, y_t) to update the model to obtain L_{t+1}. Several methods that cope with concept drift use an explicit drift detection algorithm (Wang et al., 2003a; Beyene et al., 2015; Street and Kim, 2001). One of the most used detector which is used as a benchmark in new detection methods is the Drift Detection Method (DDM) (Gama et al., 2004). It is a method that controls the trace of the on-line error by modeling the classifi-

cation errors with a Binomial distribution. Some extensions to DDM have been developed to improve its performance. Fast Hoeffding Drift Detection Method (FHDDM) (Pesaranghader and Viktor, 2016) uses a sliding window and Hoeffding's inequality to detect drifting points earlier, experimental results also show that the number of false positive and false negatives is minor (Pesaranghader and Viktor, 2016; Pesaranghader et al., 2017). Other methods extend FHDDM, however, these require additional parameters.

3. AutoML scenario and proposed methods

As previously mentioned we consider a simplified LML - AutoML with a concept drift scenario, which is precisely the scenario considered in the forthcoming AutoML3[2] challenge. The aim is assessing the robustness of methods to concept drift and its lifelong learning capabilities. It is assumed that there is an initial training (labeled) dataset available, and multiple batches of test data. Also, we assume that data are temporally dependent and are subject to an underlying form of the concept drift phenomenon. For the LML - AutoML evaluation, datasets will be split into sequential batches so that the lifelong scenarios can be evaluated: the test data will come in the form of sequential batches of data, each batch needs to be predicted at first, then the target values will be revealed and thus become a batch of new training data. From the model predictions and the revealed target values of a specific test batch, its performance can be evaluated. The performance of the model is given by taking the average across batches. Figure 1 illustrates the considered scenario.

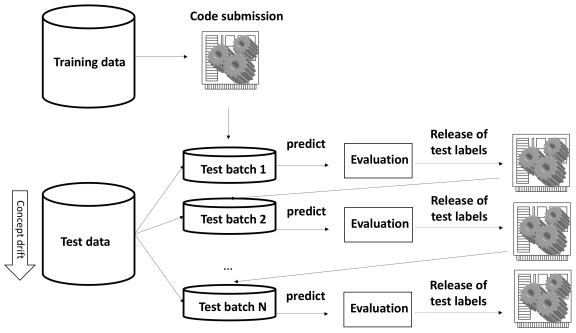

Figure 1: Evaluation scenario considered in the AutoML3 challenge.

3.1. Auto-sklearn

Auto-sklearn is an AutoML solution that has succeed in recent academic competitions (Feurer et al., 2015; Guyon et al., 2016, 2015). In our opinion, it is THE state of the art on AutoML and for that reason we considered it for our study. Auto-Skelearn is implemented in scikit-learn (Pedregosa et al., 2011), it initially comprised 15 classification algorithms, 14 preprocessing methods, and 4 data preprocessing methods. Similarly to Auto-WEKA (Thornton et al., 2013), Feurer et al. approach AutoML as a Combined Algorithm Selection and Hyperparameter (CASH) or full model selection problem (Feurer et al., 2015; Escalante et al.,

2. https://www.4paradigm.com/competition/nips2018

2009), which they tackle using SMAC, a three-based Bayesian optimization method. There are two key components that make Auto-Sklearn so competitive. The first is based on *meta-learning*, complementary to Bayesian optimization, it is used to warmstart the optimization by quickly suggesting instantiations of a framework that are likely to perform well. The meta-learning was done in an off-line phase where 38 meta-features were learned from 140 OpenML datasets (Vanschoren et al., 2014). The second feature is the automated ensemble construction of models evaluated during optimization, when finding the best model instead of discarding the rest of the models found in the Bayesian optimization process, Feurer, et. al. store them and then build and ensemble using a greedy *ensemble selection* algorithm.

3.2. Proposed method

Auto-Sklearn was modified with basic mechanisms for allowing it to cope with incoming data as depicted in Figure 1. The overall proposed procedure is shown in Algorithm 1. With a first batch, an initial ensemble model is learned by using Auto-Sklearn. With the generated model predictions are made for the next batch. After the predictions are made, feedback for this batch is received (i.e., ground truth), using a drift detector these predictions are diagnosed to determine if drift has happened. If the detector triggers the drift alarm, the current ensemble model is adapted by using one of the mechanisms described below. After this, the system receives the next batch and the process is repeated until no batches are left. In the rest of this section we describe the components of the proposed method, namely: the drift detector and the proposed adaptation mechanisms.

Data: $D(X, y)$ examples
Take a batch D'_t of size n, $D'_t(X', y') \in D(X, y)$; $T_t \leftarrow$ learn a model with auto-sklearn using D'_t
while *there is data in* D **do**
  Take next batch D'_{t+1}; $\hat{y} \leftarrow$ Make predictions with T_t; **for** $y_j \in \hat{y}$ **do**
    drift_detected = Detector($y_j == y'_j \in y'$) //drift will be detected with model performance
  end
  if *drif_detected* **then**
    $T_{t+1} = adapt(T, D'_{t+1})$; Detector.reset(); $t = t + 1$
  else
    $T_{t+1} = T_t$; $t = t + 1$
  end
end

Algorithm 1: Drift adaption schema for Auto-sklearn

3.2.1. Drift detector

In a strict LML - AutoML context, where no human intervention is expected, selecting a drift detection algorithm is not trivial, different methods perform differently depending on the type of drift in the data: incremental, abrupt, reoccurring, gradual. Some of these methods also require parameter tuning to achieve better results. For this experiment, we decided to use a particular drift detector with default hyper-parameters to make the problem more manageable and postpone to future work the automatic selection of this type of models. Specifically, the Fast Hoeffding Drift Detection Method (FHDDM) method was selected because according to experiments in (Pesaranghader et al., 2017) it was found that in many

datasets its performance can be compared to the state-of-the-art. The implementation of FHDDM in Tornado (Pesaranghader et al., 2017) was integrated with AutoSKlearn.

3.2.2. Model adaptation methods

Three approaches were implemented for adapting the model generated by Auto-Sklearn, a global replacement strategy where the initial model found by Auto-Sklearn is retrained with new examples and two variants of model management strategies.

Model replacement. A traditional concept drift adaptation where the model is globally replaced with a new one. Taking advantage of the Auto-Sklearn improvements, the model is not trained from scratch. Having the meta-features learned with the meta-learning process from Auto-Sklearn, these are used for the creation of the new model, which is a new ensemble with the best models found by the SMAC process, the learning for the experimentation was performed storing all data received instead of only the current batch.

Model management. Two variants of this formulation were considered. The first strategy assumes the data is generated by a mixture distribution (that formed by previous batches and the current one) and the ensemble weights from the initially learned ensemble are *updated* with new data, either: (1) using the latest batch or (2) using all the stored data up to the current batch, we call these variants WU-latest and WU-all, respectively. Creators of AutoSKlearn (Feurer et al., 2015) found that uniformly weighted ensembles did not work well. Instead, a greedy *ensemble selection* algorithm was used to build the ensemble. Such same algorithm is used in this work to update the weights of each model under the model management variants. With the new data every model performance is computed again and the ensemble is rebuild with new information.

In a second model management variant called *Add New*, new models are learned with all the stored data, meta-learning process is also skipped in this case, and using auto-sklearn ensemble construction methods these new models are integrated in the initial ensemble. However, the implementation of this last strategy did not work as expected since the original design of auto-sklearn does not consider adding new models to the ensemble so a new model selection algorithm for the ensemble construction ought to be implemented.

4. Experiments and results

4.1. Data

For the empirical evaluation we used benchmark data form both concept drift and the AutoML fields, where the AutoML data sets considered are known to incorporate temporal dependencies across instances. The considered data sets are described in Table 1.

Concept drift data sets have been widely used for the evaluation of drift detection techniques. *Chess* comprises 2 years of chess games collected from `chess.com`, the task is to predict whether a player will win or lose a match given players and game setup features, the player increases its abilities through time and faces more skilled players, this is where the concept drift is expected. In the *Poker* dataset each instance is an example of a hand having five playing cards drawn from a standard deck. Each card is described by two attributes (suit and rank) for a total of ten attributes. The class predicts the poker hand, 10 classes are considered. Since the poker hands were generated in order (varying suits and ranks in

Table 1: Datasets considered for experimentation.

Concept drift datasets			
Dataset	**instances**	**attributes**	**Reference**
Chess	503	8	(Žliobaite, 2010)
Poker	100,000	10	(Olorunnimbe et al., 2015)
Electricity	45,312	8	(Baena-García et al., 2006)
Stagger	70,000	3	(Gama et al., 2014)
AutoML2 challenge data sets			
Dataset	**instances**	**attributes**	**Reference**
PM	49,964	89	(Guyon et al., 2018)
RH	60,042	76	(Guyon et al., 2018)
RI	57,306	113	(Guyon et al., 2018)
RL	56,209	22	(Guyon et al., 2018)
RM	55,239	89	(Guyon et al., 2018)

order), the concept drifts multiple times. The *electricity* market dataset was collected from the Australian New South Wales Electricity Market. Each instance represents the market state in a period of 30 minutes. The task in this dataset is to predict if the market prices are going up or down for the next period of time. Finally, a synthetic Stagger dataset was generated to test the methods, three drift points are artificially induced were the concept changes abruptly. According to previous work, we used accuracy as evaluation measure for these datasets.

AutoML data sets comprise undisclosed data from click user data, these datasets were used in the AutoML2 challenge (Guyon et al., 2018). Although the data has been keep confidential, it is known that there is a temporal dependency across instances. We expect the proposed methods to capture these dependencies and improve the performance of a straightforward AutoML technique. We know *RH* and *RI* were collected from applications in startup phase, and behavior changed dramatically during that time. *RL* was collected from content recommendation applications, the items were user generated contents, there were a certain proportion of new contents generated each day, the user preferences of the items were changed along time. *PM* and *RM* were collected from a mature recommender system in different time periods, the items were cars, the candidate set of cars was relatively stable along time, the user preferences were also relatively stable compared to other datasets. For these datasets we used the normalized area under the ROC curve as evaluation measure.

4.2. Results

For each of the considered datasets the following setup was considered. Training partitions were used to obtain an AutoSKlearn model. For concept drift datasets, the data was divided into batches of equal size, and the first batch was used as training set. For the AutoML datasets the predefined training partitions were used for generating the initial model. After that, the test set was processed in batches. The actual model was used to predict labels for the batch t and performance was estimated, after that, test labels for this batch were made available to the method. Then we applied the drift detector. If drift was detected, the methods described in Section 3 were applied to adapt the current model, which was then

used for batch $t+1$. In experiments, we compare the performance of the initial AutoSKlearn model (*Base*) with the replacement, WU-all, WU-batch and the Add New methods.

Table 2 reports the average (across batches) performance for the concept drift datasets. Regarding the real datasets, in all but one configuration (*Poker* dataset with WU-latest method) the drift aware AutoML solutions outperformed the base model. As it can be expected, replacing the initial model with a new model obtained by applying AutoSKlearn with all available data obtained the best performance. Model replacement, however, is the most computationally expensive solution. A competitive alternative is the WU-all method, in which ensemble weights are updated by using all of the available data. In our experiments, WU-all method was in the worse case about 5 times faster than the replacement technique. The speedup was more noticeable in large datasets.

Interestingly, most drift aware methods failed with the synthetic data set (*Stagger*), this was due in part to the abrupt drift present in data: synthetic drift is simulated by inverting the concept (negative instances are considered as positive ones). WU-all and WU-latest models only consider models built before the drift (i.e. with the rule before the drift) and thus fail to classify correctly after drift detected. The Add new method also tries to incrementally adjust to changes, but AutoSKlearn fails to find an ensemble that classifies correctly both instances before and after the drift. On the other hand, the replacement method builds models from scratch after drift is detected and is able to improve the accuracy of the ensemble.

Table 2: Results on benchmark data.

Method	**Electricity**	**Poker**	**Chess**	**Stagger**	**Rank**
Base	67.15	67.97	38.23	54.09	4.5
Replacement	**76.44**	**90.38**	**58.13**	**78.81**	1
WU-all	70.23	76.89	52.62	54.09	2.75
WU-latest	67.95	67.49	53.24	54.09	3.5
Add new model	67.47	74.98	47.28	54.14	3.25

Figure 2 shows the per-batch improvement of AutoML drift aware methods vs. using the initial model. The figure shows a somewhat erratic behavior at times, although, in general, these plots confirm the results presented in Table 2. It can be seen from these plots that the WU-batch method is very sensitive to the batch used to adjust weights (see rightmost top plot). The Replacement and WU-all strategy consistently outperform the base model in the real data sets, and for the synthetic dataset (rightmost bottom) all methods but replacement perform poorly.

Table 3 shows the results obtained in the AutoML2 challenge data sets, performance is reported in terms of the normalized AUC. For *PM* and *RM* datasets the drift detector did not detect drift in any batch, and therefore the base model was used for making predictions in all the batches. This could be due to the maturity of the application from which *PM* and *RM* were collected from. The drift of concept evolves slowly and it cannot be captured by the batch size. On the other hand, for datasets collected from startup businesses or UGC recommender systems, due to the changing nature of the applications, for *RH, RI* and *RL*, at least one of the adaptive AutoML methods outperformed considerably the base

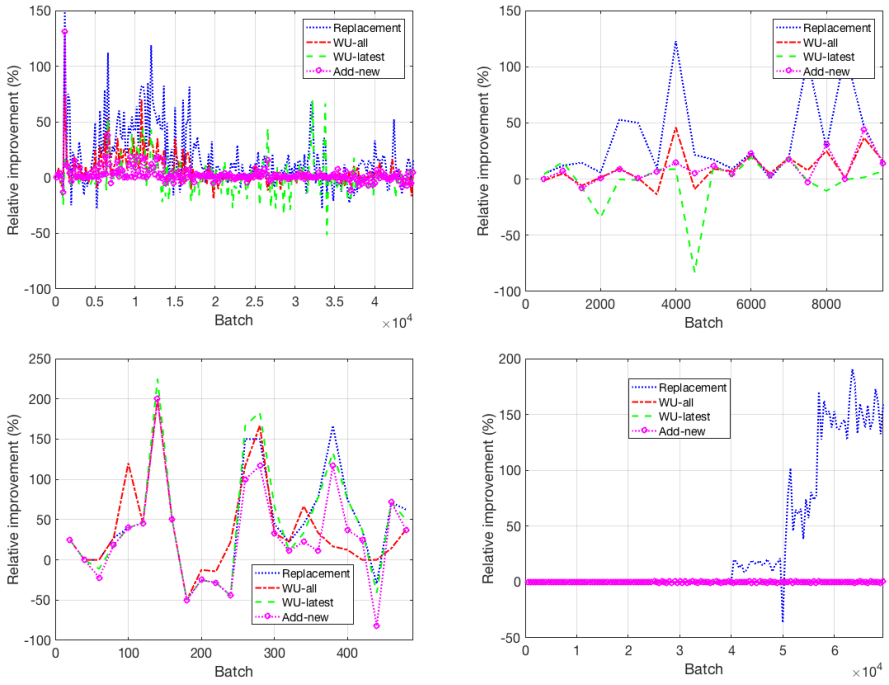

Figure 2: Performance of drift aware AutoML variants. From top to bottom and left to right results for: Electricity, Poker, Chess and Stagger are shown, respectively.

performance. For *RL*, the Replacement method improved relatively the performance of the base model by 47%. For *RI* the WU-latest method improved the initial performance by more than 60%. Whereas for *RH* all of the adaptive AutoML variants outperformed the reference performance; the highest relative improvement being of 73% obtained by the WU-all strategy.

Table 3: Results on data from the AutoML2 challenge.

Method	**PM***	**RH**	**RI**	**RL**	**RM***
Base	0.433	0.192	0.299	0.340	0.264
Replacement	0.433	0.197	0.092	**0.478**	0.264
WU-all	0.433	**0.370**	0.199	0.212	0.264
WU-latest	0.433	0.270	**0.450**	0.405	0.264
Add new	0.433	0.298	0.184	0.277	0.264

Although more experiments are needed, the results in AutoML2 data suggest/confirm that each dataset presents a different variant of the concept drift phenomenon. *RL* seems to follow the behavior of datasets in Table 2. *RH* seems to exhibit a slowly progressing drift,

where WU-all takes advantage. While *RI* seems to show a drift aligned with the concept drift detector (hence the WU-latest method is able to find quite useful weights).

5. Conclusions

AutoML is an increasingly trending topic because of its relevance given the amount of data being generated nowadays. Although very effective AutoML methods have been proposed so far, to the best of our knowledge all of them focus on the assumption of static (iid) data. However, on-line data is constantly changing and therefore, static solutions may not be useful for evolving data streams. This paper described preliminary experimentation on the evaluation of the drift adaptive capabilities of a state of the art AutoML solution. To the best of our knowledge this is the first work dealing with AutoML in the presence of drift. Experimental results in benchmark data confirm the usefulness of the proposed mechanisms for dealing with drift. Although our results are far from being conclusive, they bring some light into the performance of AutoML for evolving data streams.

References

Manuel Baena-García, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, Ricard Gavaldà, and Rafael Morales-Bueno. Early drift detection method. 2006.

Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

Yoshua Bengio. Gradient-based optimization of hyperparameters. *Neural computation*, 12 (8):1889–1900, 2000.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*, pages 2546–2554, 2011.

Ayne A. Beyene, Tewelle Welemariam, Marie Persson, and Niklas Lavesson. Improved concept drift handling in surgery prediction and other applications. *Knowl. Inf. Syst.*, 44 (1):177–196, July 2015. ISSN 0219-1377. doi: 10.1007/s10115-014-0756-9. URL http://dx.doi.org/10.1007/s10115-014-0756-9.

Albert Bifet and Ricard Gavalda. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*, pages 443–448. SIAM, 2007.

Albert Bifet, Geoff Holmes, and Bernhard Pfahringer. Leveraging bagging for evolving data streams. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 135–150. Springer, 2010.

Hamparsum Bozdogan. Model selection and akaike's information criterion (aic): The general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

Zhiyuan Chen, Nianzu Ma, and Bing Liu. Lifelong learning for sentiment classification. *arXiv preprint arXiv:1801.02808*, 2018.

Hugo Jair Escalante, Manuel Montes, and Luis Enrique Sucar. Particle swarm model selection. *J. Mach. Learn. Res.*, 10:405–440, June 2009. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1577069.1577084.

Geli Fei, Shuai Wang, and Bing Liu. Learning cumulatively to become more knowledgeable. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1565–1574. ACM, 2016.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2962–2970. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM Comput. Surv.*, 46(4):44:1–44:37, March 2014. ISSN 0360-0300. doi: 10.1145/2523813. URL http://doi.acm.org/10.1145/2523813.

João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In Ana L. C. Bazzan and Sofiane Labidi, editors, *Advances in Artificial Intelligence – SBIA 2004*, pages 286–295, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

Isabelle Guyon, Kristin Bennett, Gavin C. Cawley, Hugo Jair Escalante, Sergio Escalera, Tin Kam Ho, Núria Macià, Bisakha Ray, Mehreen Saeed, Alexander R. Statnikov, and Evelyne Viegas. Design of the 2015 chalearn automl challenge. In *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-17, 2015* Guyon et al. (2015), pages 1–8. doi: 10.1109/IJCNN.2015.7280767. URL https://doi.org/10.1109/IJCNN.2015.7280767.

Isabelle Guyon, Imad Chaabane, Hugo Jair Escalante, Sergio Escalera, Damir Jajetic, James Robert Lloyd, Núria Macià, Bisakha Ray, Lukasz Romaszko, Michèle Sebag, Alexander R. Statnikov, Sébastien Treguer, and Evelyne Viegas. A brief review of the chalearn automl challenge: Any-time any-dataset learning without human intervention. In *Proceedings of the 2016 Workshop on Automatic Machine Learning, AutoML 2016, co-located with 33rd International Conference on Machine Learning (ICML 2016), New York City, NY, USA, June 24, 2016*, pages 21–30, 2016. URL http://jmlr.org/proceedings/papers/v64/guyon_review_2016.html.

Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michele Sebag, Alexander Statnikov, Wei-Wei Tu, and Evelyne Viegas. Analysis of the automl challenge series 2015-2018. In Frank Hutter et al., editor, *AutoML, Springer Series on Challenges in Machine Learning (to appear)*. 2018.

Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 97–106. ACM, 2001.

Frank Hutter. *Automated configuration of algorithms for solving hard computational problems*. PhD thesis, University of British Columbia, 2009.

Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.

Mario A Muñoz, Laura Villanova, Davaatseren Baatar, and Kate Smith-Miles. Instance spaces for machine learning classification. *Machine Learning*, 107(1):109–147, 2018.

M Kehinde Olorunnimbe, Herna L Viktor, and Eric Paquet. Intelligent adaptive ensembles for data stream mining: a high return on investment approach. In *International Workshop on New Frontiers in Mining Complex Patterns*, pages 61–75. Springer, 2015.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. URL http://dl.acm.org/citation.cfm?id=2078195.

Ali Pesaranghader and Herna L. Viktor. Fast hoeffding drift detection method for evolving data streams. In Paolo Frasconi, Niels Landwehr, Giuseppe Manco, and Jilles Vreeken, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 96–111, Cham, 2016. Springer International Publishing.

Ali Pesaranghader, Herna Viktor, and Eric Paquet. Reservoir of diverse adaptive learners and stacking fast hoeffding drift detection methods for evolving data streams. *CoRR*, abs/1709.02457, 2017. URL http://arxiv.org/abs/1709.02457.

Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017.

André Luis Debiaso Rossi, André Carlos Ponce de Leon Ferreira, Carlos Soares, Bruno Feres De Souza, et al. Metastream: A meta-learning based method for periodic algorithm selection in time-changing data. *Neurocomputing*, 127:52–64, 2014.

Jeffrey C Schlimmer and Richard H Granger. Beyond incremental processing: Tracking concept drift. In *AAAI*, pages 502–507, 1986.

Daniel Silver and Robert Mercer. Selective functional transfer: Inductive bias from related tasks. In *IASTED International Conference on Artificial Intelligence and Soft Computing (ASC2001)*, pages 182–189, 2001.

Daniel L. Silver and Qiang Yang amd Lianghao Li. Lifelong machine learning systems: Beyond learning algorithms. In *Lifelong Machine Learning: Papers from the 2013 AAAI Spring Symposium*, pages 49–55, 2013.

Daniel L Silver and Robert E Mercer. The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. In *Learning to learn*, pages 213–233. Springer, 1996.

Daniel L Silver and Robert E Mercer. The task rehearsal method of life-long learning: Overcoming impoverished data. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 90–101. Springer, 2002.

W. Nick Street and YongSeog Kim. A streaming ensemble algorithm (sea) for large-scale classification. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 377–382, New York, NY, USA, 2001. ACM. ISBN 1-58113-391-X. doi: 10.1145/502512.502568. URL http://doi.acm.org/10.1145/502512.502568.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw bayesian optimization. *arXiv preprint arXiv:1406.3896*, 2014.

Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 847–855, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2174-7. doi: 10.1145/2487575.2487629. URL http://doi.acm.org/10.1145/2487575.2487629.

Sebastian Thrun and Tom M Mitchell. Lifelong robot learning. *Robotics and autonomous systems*, 15(1-2):25–46, 1995.

Jan N van Rijn, Geoffrey Holmes, Bernhard Pfahringer, and Joaquin Vanschoren. Algorithm selection on data streams. In *International Conference on Discovery Science*, pages 325–336. Springer, 2014.

Jan N van Rijn, Geoffrey Holmes, Bernhard Pfahringer, and Joaquin Vanschoren. Having a blast: Meta-learning and heterogeneous ensembles for data streams. In *Data Mining (ICDM), 2015 IEEE International Conference on*, pages 1003–1008. IEEE, 2015.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explor. Newsl.*, 15(2):49–60, June 2014. ISSN 1931-0145. doi: 10.1145/2641190.2641198. URL http://doi.acm.org/10.1145/2641190.2641198.

Haixun Wang, Wei Fan, Philip S. Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 226–235, New York, NY, USA, 2003a. ACM. ISBN 1-58113-737-0. doi: 10.1145/956750.956778. URL http://doi.acm.org/10.1145/956750.956778.

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235. AcM, 2003b.

Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine learning*, 23(1):69–101, 1996.

Indre Žliobaite. Change with delayed labeling: When is it detectable? In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 843–850. IEEE, 2010.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.